

# Mettre en place simplement la sécurité de ces applications avec Keycloak

Lilian BENOIT  
20 juin 2018



# Qui ?

- Lilian BENOIT
- JUG Leader du BordeauxJUG
- Tech Leader d'une ESN Bordelaise : IMC
- 18 ans dans l'informatique



@Lilian\_Benoit



lilian-benoit



# Pourquoi déléguer sa sécurité ?



# Pourquoi déléguer sa sécurité ?

Réutilisation / Economies



# Pourquoi déléguer sa sécurité ?

Réutilisation / Economies



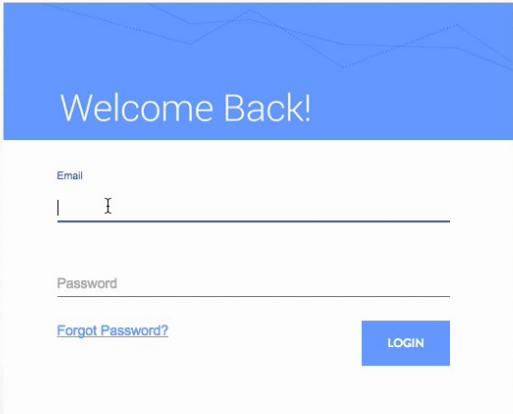
**Employee Logon**

Username:

Password:

# Pourquoi déléguer sa sécurité ?

## Réutilisation / Economies



Welcome Back!

Email

Password

[Forgot Password?](#)



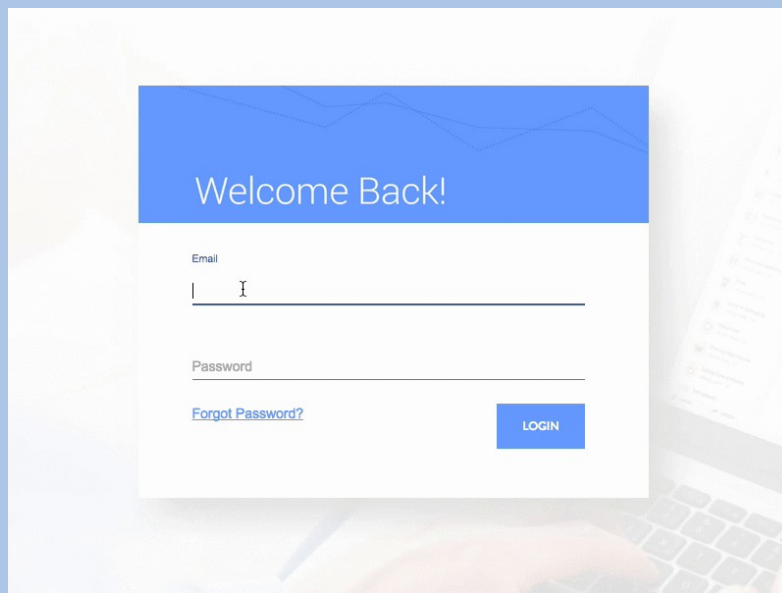
**Employee Logon**

Username:

Password:

# Pourquoi déléguer sa sécurité ?

## Réutilisation / Economies



UNITED STATES  
DEPARTMENT OF LABOR

SEARCH

A to Z Index | En Español | Contact Us | FAQs | About OSHA

OSHA

OSHA QuickTakes Newsletter RSS Feeds Print This Page Text Size

Occupational Safety & Health Administration We Can Help

Home Workers Regulations Enforcement Data & Statistics Training Publications Newsroom Small Business OSHA

What's New | Offices

### OSHA's Injury/Illness Data Collection

**REGISTRATION:**

In order to submit your injury and illness data to OSHA, you must first register by completing the form below. After submission of the registration form, a confirmation email will be sent to the email address provided. This email will contain your ID and password to log into the data submission dashboard.

Please provide the following information (\* = required)

* First Name:	<input type="text"/>
* Last Name:	<input type="text"/>
* Title:	<input type="text"/>
* Company Name:	<input type="text"/>
* Address 1:	<input type="text"/>
* Address 2:	<input type="text"/>
* City:	<input type="text"/>
* State:	<input type="text"/>
* Zip:	<input type="text"/>
* Phone:	<input type="text"/>
* Ext:	<input type="text"/>
* Email:	<input type="text"/>
* Confirm Email:	<input type="text"/>
* Select a Security Question:	<input type="text"/>
* Answer:	<input type="text"/>

Feedback | Disclaimer

U.S. Department of Labor | Frances Perkins Building, 200 Constitution Ave., NW, Washington, DC 20202  
www.dol.gov | Telephone: 1-866-4-USA-DOOL | TTY: 1-877-889-5627 | Contact Us

# Pourquoi déléguer sa sécurité ?

Nous ne sommes pas des experts





# Comment cela fonctionne ?

- L'application redirige vers un fournisseur d'identité
- L'utilisateur s'identifie auprès du fournisseur, qui valide la phase d'identification
- L'application utilise les informations du fournisseur



# Coté protocole

- OAuth 2.0
- OpenID Connect 1.0
- SAML 2.0



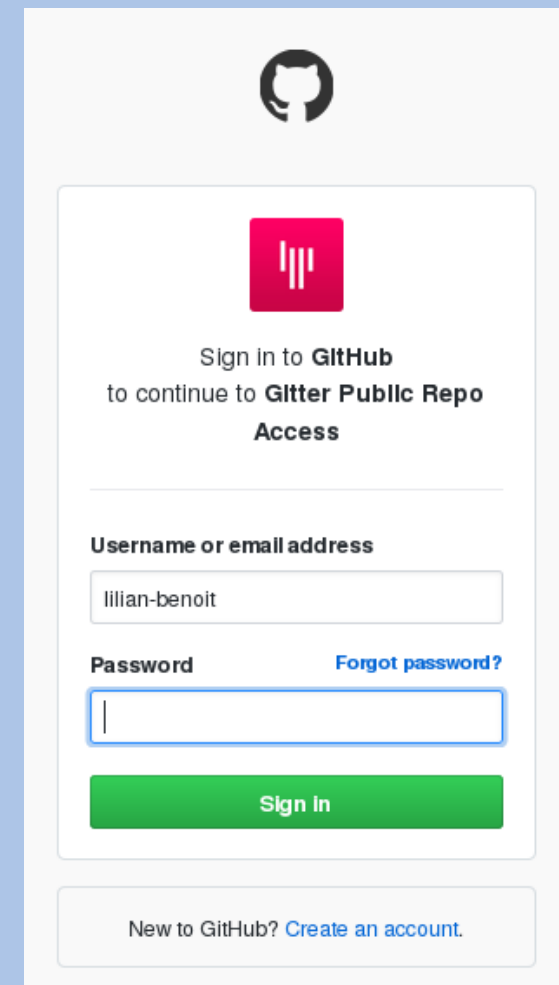
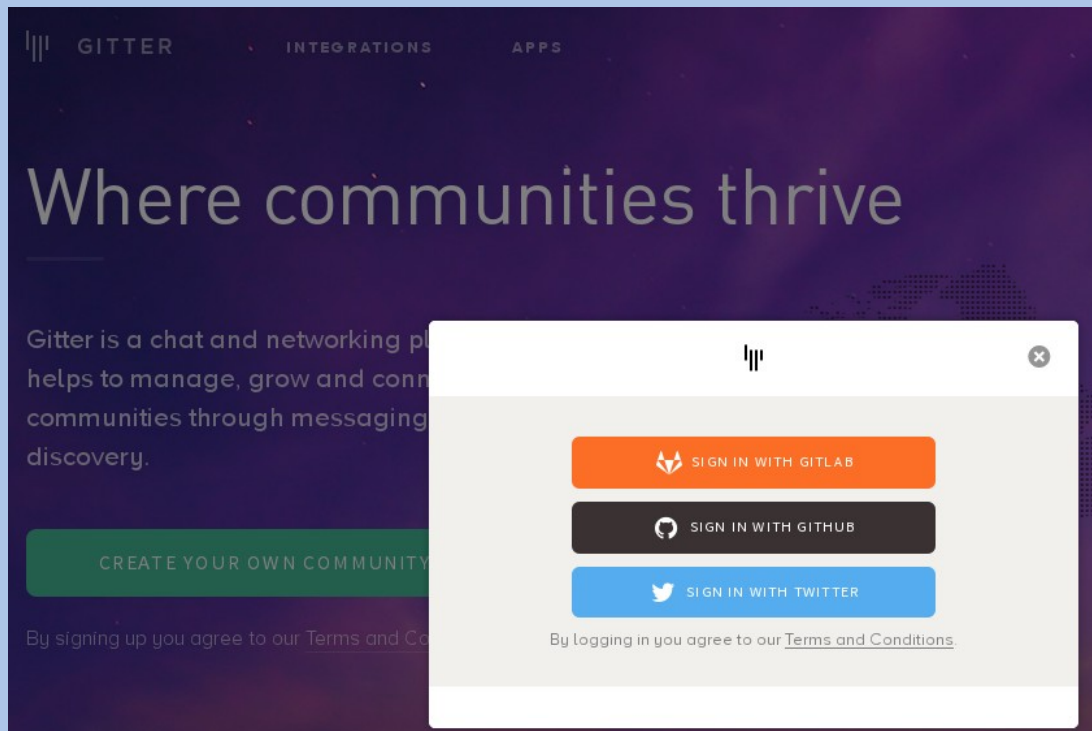
# OAuth 2.0

- Protocole de délégation d'autorisation.



# OAuth 2.0 – Coté Client

- Lilian utilise son compte GitHub pour se connecter à Gitter.im.

A screenshot of the GitHub login form. At the top is the GitHub logo. Below it is a red square with the Gitter logo. The text says "Sign in to GitHub to continue to Gitter Public Repo Access". There are two input fields: "Username or email address" with the value "lilian-benoit" and "Password". A link "Forgot password?" is next to the password field. A green "Sign In" button is at the bottom. At the very bottom, it says "New to GitHub? Create an account."

# OAuth 2.0 – Coté Client

- Lilian autorise Gitter.im à appeler l'API Github en son nom

## Applications

Installed GitHub Apps

Authorized GitHub Apps

Authorized OAuth Apps

You have granted **4 applications** access to your account.

Sort ▼

Revoke all



### Gitter Public Repo Access

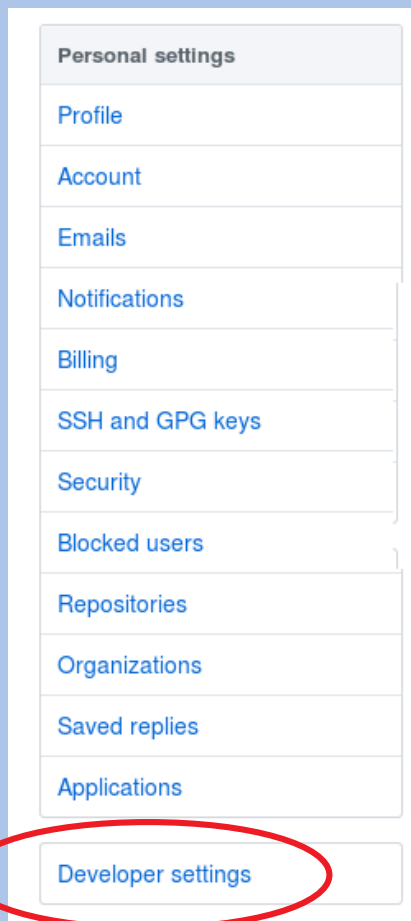
Last used within the last 2 weeks · Owned by [gitterHQ](#)

Revoke



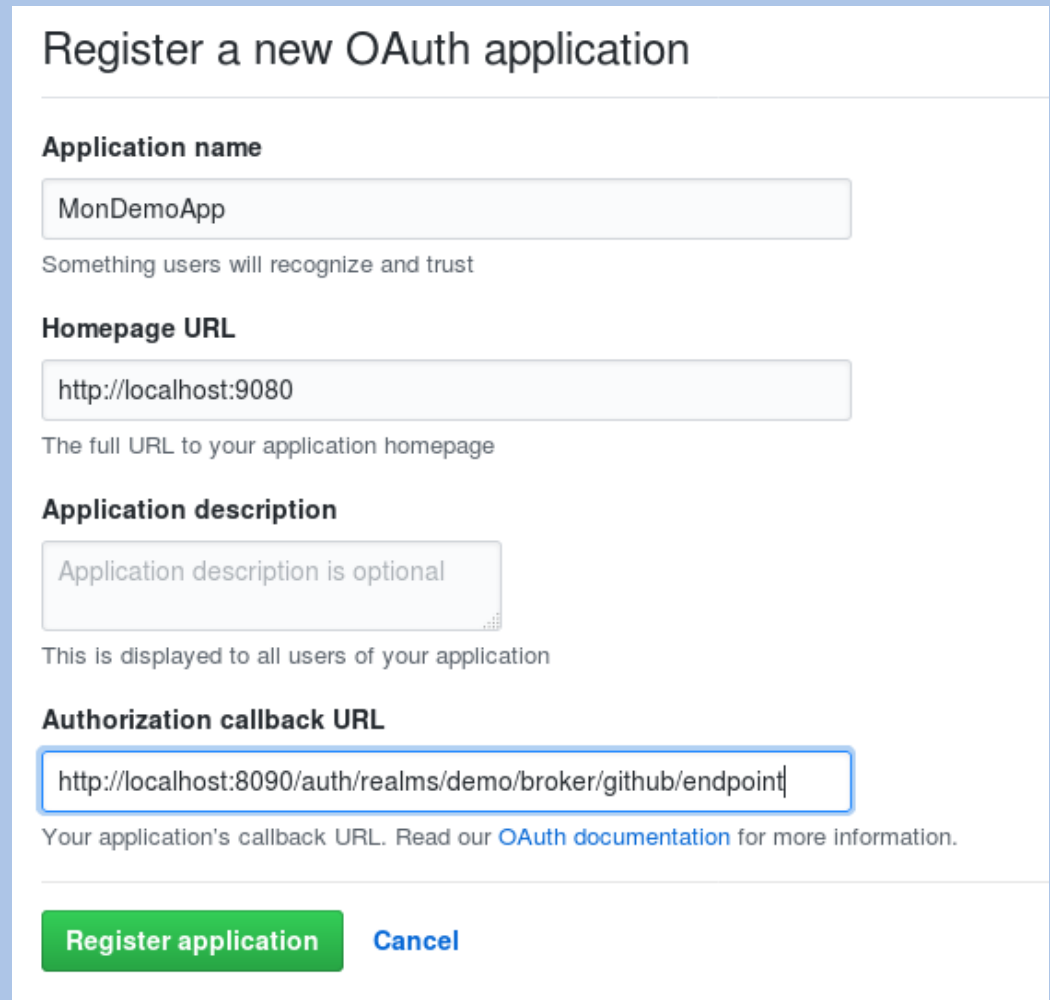
# OAuth 2.0 – Coté Application

- Il est nécessaire de déclarer l'application



Personal settings

- Profile
- Account
- Emails
- Notifications
- Billing
- SSH and GPG keys
- Security
- Blocked users
- Repositories
- Organizations
- Saved replies
- Applications
- Developer settings**



## Register a new OAuth application

**Application name**

Something users will recognize and trust

**Homepage URL**

The full URL to your application homepage

**Application description**

This is displayed to all users of your application

**Authorization callback URL**

Your application's callback URL. Read our [OAuth documentation](#) for more information.

**Register application** [Cancel](#)

# OAuth 2.0 – Coté Application

## MonDemoApp



**lilian-benoit** owns this application.

[Transfer ownership](#)

You can list your application in the [GitHub Marketplace](#) so that other users can discover it.

[List this application in the Marketplace](#)

**0** users

**Client ID**

b9076badd4e7b83b502

**Client Secret**

82e7a1e650524421c4544185970c26dcabd87c25

[Revoke all user tokens](#)

[Reset client secret](#)



# OAuth 2.0 – le déroulé (1/4)

L'utilisateur clique sur le bouton  
« Connecter avec son compte GitHub »





# OAuth 2.0 – le déroulé (2/4)

L'application redirige vers github.com

GET https://localhost:8090/link-account/github.com HTTP/1.1

HTTP/1.1 302 Found

Location: https://github.com/login/oauth/authorize \

?client\_id=b9076baddd4e7b83b502 \

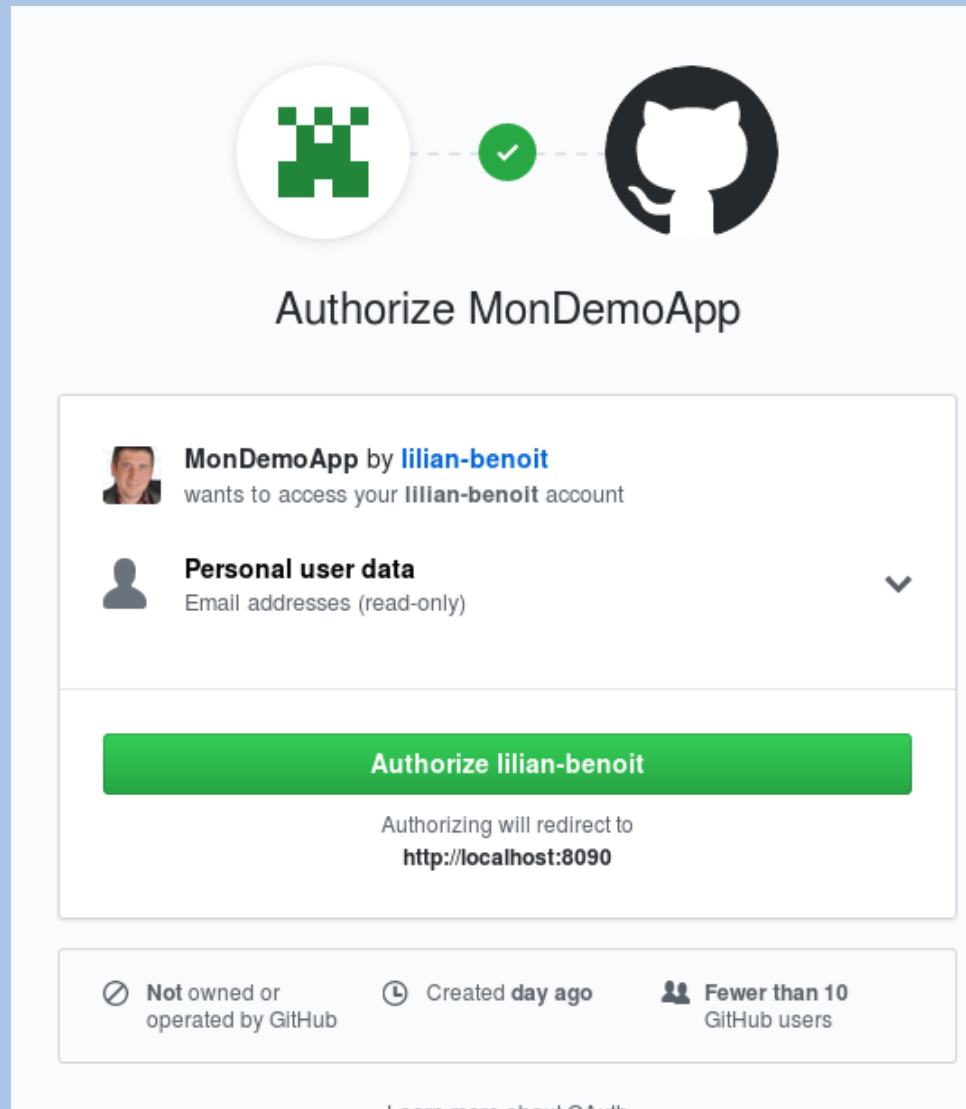
&redirect\_uri=http://localhost:8090/auth/realms/demo/broker/github/endpoint \

&state=1efg \

&scope=user,repo



# OAuth 2.0 – le déroulé (3/4)



# OAuth 2.0 – le déroulé (4/4)

Github.com redirige vers l'URL de callback

GET https://github.com/login/oauth/authorize... HTTP/1.1

HTTP/1.1 302 Found

Location: http://localhost:8090/auth/realms/demo/broker/github/endpoint/  
?code=f8a24125b6a45b05e81a&state=1efg



# OAuth 2.0 – usage (1/2)

L'application échange le code contre un token.

```
curl -H "Accept: application/json" \  
  --data "client_id=b9076baddd4e7b83b502" \  
  --data "client_secret=xxxxxxxxxxxxx" \  
  --data "code=b9076baddd4e7b83b502" \  
  --data "state=1efg" \  
  https://github.com/login/oauth/access_token  
  
{  
  "access_token":"34a8a4e387f9c68f611183497c0969d",  
  "token_type":"bearer",  
  "scope":"repo,user"  
}
```



# OAuth 2.0 – usage (2/2)

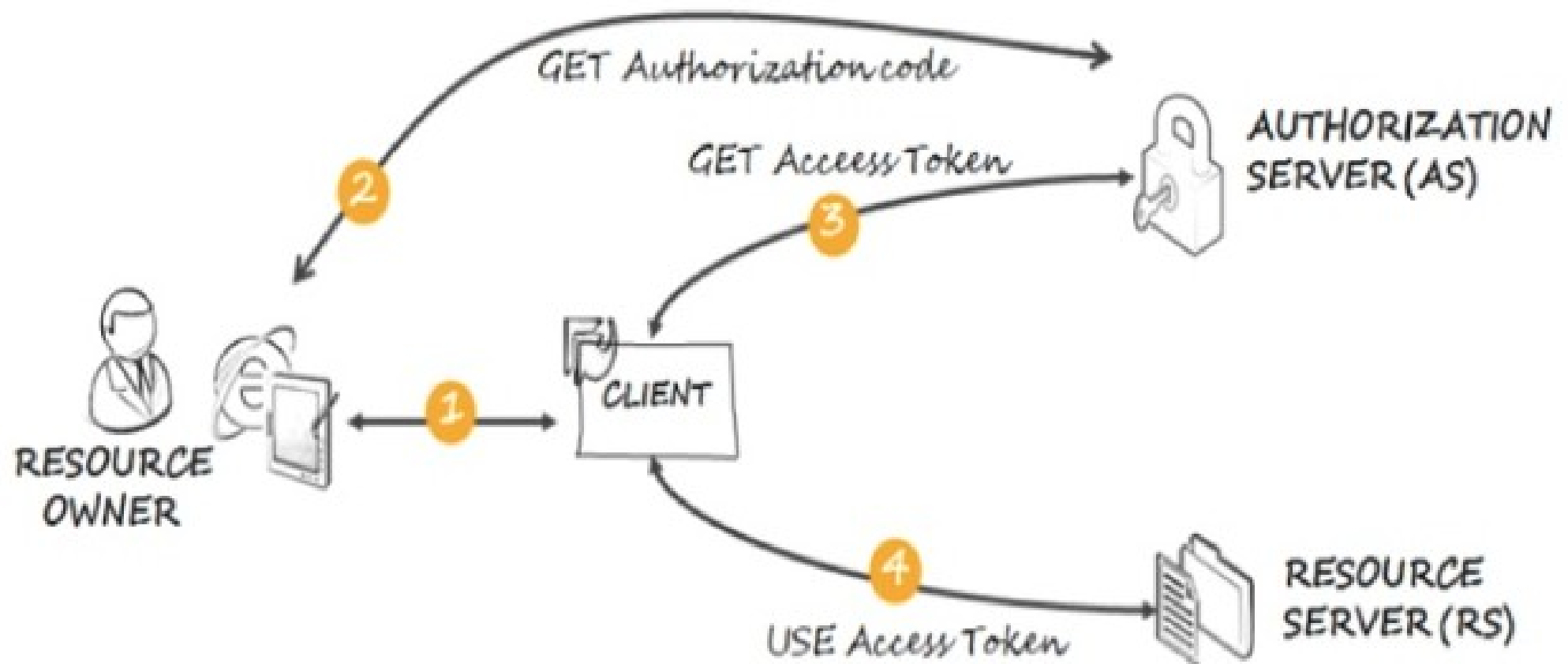
L'application peut utiliser le token  
afin d'utiliser l'API github.

```
curl -H "Authorization: Bearer 34a8a4e387f9c68f611183497c0969d" -H "Accept: application/json" \
https://api.github.com/user
```

```
{
  "login": "lbenoit",
  "id": 15512,
  "avatar_url": "https://avatars1.githubusercontent.com/u/15512?v=4",
  "gravatar_id": "",
  "url": "https://api.github.com/users/lbenoit",
  "html_url": "https://github.com/lbenoit",
  "name": "Lilian BENOIT",
  "location": "Bordeaux, France",
  ....
}
```



# OAuth 2.0 – en résumé



# OAuth 2.0 : les limites

- Uniquement une délégation, (pas d'information sur l'utilisateur)
- Si besoin, appel d'un API non standard
- Plusieurs applications, chacun doit appeler l'API du fournisseur
- Pas de déconnexion globale (Single Sign Out)



# JSON Web Token (JWT)

Format de transport de données signé

« Base64(Header) ». « Base64(Payload) ». « Base64(Signature) »





# JWT - Exemple

## Encoded

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXV  
CJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwib  
mFtZSI6IkpvaG4gRG9lIiwiaWF0IjOn  
RydWV9.TJVA950rM7E2cBab30RMHrHDcE  
fxjoYZgeFONFh7HgQ
```

## Decoded

### HEADER:

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}
```

### PAYLOAD:

```
{  
  "sub": "1234567890",  
  "name": "John Doe",  
  "admin": true  
}
```

### VERIFY SIGNATURE

```
HMACSHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
    
) ☐ secret base64 encoded
```



# OpenID Connect

OpenID Connect 1.0 est une simple couche d'identité au dessus du protocole OAuth 2.0

Développé par OpenID Foundation

OpenID Connect core: Novembre 2014



# OpenID Connect

- OpenID Connect = OAuth 2.0 + IDToken + JWT
- Discovery
- Front channel logout
- Back channel logout
- ...



# ID Token

L'ID Token est fourni avec l'Access Token

```
{  
  "access_token": "ya29.GlsQBRz-E5HEoo6KeQNsZQK-XazzgybwWXgOEU1jHu3y...",  
  "token_type": "Bearer",  
  "expires_in": 3600,  
  "id_token": "eyJhbGciOiJSUzI1NiIsImtpZCI6IjBIMTgzMzE1NzY2NDgyNGZOTc..."  
}
```



# ID Token

Un ID Token possède des attributs ayant des noms normalisées.

Member	Type	Description
sub	string	Subject - Identifier for the End-User at the Issuer.
name	string	End-User's full name in displayable form including all name parts.
given_name	string	Given name(s) or first name(s) of the End-User. Note that in some cultures, names may be written in reverse order.
family_name	string	Surname(s) or last name(s) of the End-User. Note that in some cultures, names may be written in reverse order.
middle_name	string	Middle name(s) of the End-User. Note that in some cultures, people do not use a middle name.
nickname	string	Casual name of the End-User that may or may not be the same as the name.
preferred_username	string	Shorthand name by which the End-User wishes to be referred to (e.g., a username or identifier). The value MUST NOT rely upon this value being unique, as discussed in <a href="#">Section 4.1.2</a> .
profile	string	URL of the End-User's profile page. The contents of this Web page SHOULD be a simple HTML document.
picture	string	URL of the End-User's profile picture. This URL MUST refer to an image that is a profile photo of the End-User suitable for displaying when describing the End-User.
website	string	URL of the End-User's Web page or blog. This Web page SHOULD contain information about the End-User.
email	string	End-User's preferred e-mail address. Its value MUST conform to the <a href="#">RFC 5322</a> syntax.
email_verified	boolean	True if the End-User's e-mail address has been verified; otherwise, the value is false. The means by which an e-mail address is verified is described in <a href="#">Section 4.1.2.6</a> .
gender	string	End-User's gender. Values defined by this specification are <a href="#">female</a> , <a href="#">male</a> , and <a href="#">other</a> . The value <a href="#">other</a> is used when the End-User's gender is not <a href="#">female</a> or <a href="#">male</a> .
birthdate	string	End-User's birthday, represented as an <a href="#">ISO 8601:2004</a> [ISO8601] date in full, extended format, consisting of the date and the time zone, or the underlying platform's date related function, providing just year and month.
zoneinfo	string	String from zoneinfo <a href="#">[zoneinfo]</a> time zone database representing the End-User's time zone.
locale	string	End-User's locale, represented as a <a href="#">BCP47</a> [RFC5646] language tag, consisting of a language and a script, separated by a dash. For example, <a href="#">en-US</a> or <a href="#">fr-CA</a> . As a compact form, the End-User's preferred language may be represented using this locale syntax as well.
phone_number	string	End-User's preferred telephone number. <a href="#">E.164</a> [E.164] is RECOMMENDED that the extension be represented using the <a href="#">E.164</a> format.
phone_number_verified	boolean	True if the End-User's phone number has been verified; otherwise, the value is false. The means by which a phone number is verified is described in <a href="#">Section 4.1.2.6</a> . The <a href="#">phone_number</a> Claim MUST be in E.164 format and any extension.
address	JSON object	End-User's preferred postal address. The value of the <a href="#">address</a> Claim MUST be a JSON object containing the following members:
updated_at	number	Time the End-User's information was last updated. Its value is a <a href="#">RFC 3339</a> [RFC3339] date and time, to the second.

# Discovery

```
curl http://localhost:8090/auth/realms/demo/.well-known/openid-configuration
```

```
{
  "issuer": "http://localhost:8090/auth/realms/demo",
  "authorization_endpoint": "http://localhost:8090/auth/realms/demo/protocol/openid-connect/auth",
  "token_endpoint": "http://localhost:8090/auth/realms/demo/protocol/openid-connect/token",
  "token_introspection_endpoint": "http://localhost:8090/auth/realms/demo/protocol/openid-connect/token/introspect",
  "userinfo_endpoint": "http://localhost:8090/auth/realms/demo/protocol/openid-connect/userinfo",
  "end_session_endpoint": "http://localhost:8090/auth/realms/demo/protocol/openid-connect/logout",
  "jwks_uri": "http://localhost:8090/auth/realms/demo/protocol/openid-connect/certs",
  "check_session_iframe": "http://localhost:8090/auth/realms/demo/protocol/openid-connect/login-status-iframe.html",
  "grant_types_supported": [
    "authorization_code",
    "implicit",
    "refresh_token",
    ...
  ]
}
```



# Refresh Token

Objectif : Garantir la cohérence des droits sans passer par la procédure initiale.

- Access Token et ID Token à durée de vie courte
- Refresh Token à durée de vie longue

```
{  
  "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldU...",  
  "expires_in": 300,  
  "refresh_expires_in": 1800,  
  "refresh_token": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJVV... ",  
  "token_type": "bearer",  
  "id_token": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldU..."  
}
```



# SAML 2.0

Protocole ouvert et standardisé pour l'échange des informations d'authentification et d'autorisation entre des entités





# SAML 2.0

Security Assertion Markup Language

Basé sur XML

Standard approuvé par le consortium 

SAML 1.0 – Novembre 2002

SAML 1.1 – Novembre 2003

SAML 2.0 – Mars 2005



# SAML 2.0 - Avantages

- Web SSO
- Autorisation basée sur les attributs
- Fédération d'identité
- Pas limité au protocole HTTP



# Installation et déploiement de Keycloak



# DEMO

Installation / Lancement

Sécurisation d'une application Spring Boot

Tour d'horizon du produit



# Les adaptateurs

Quels sont leurs rôles ?

- Gestion des Redirections (ou pas)
- Vérification des signatures
- Vérifications des Claims



# Les adaptateurs

Est-ce que la ressource est-elle soumise à une contrainte de sécurité ?

Y'a t-il un Header authorization ?



# Les adaptateurs – cas 1

Cas 1 : l'header authorization n'est pas présent

Le client est-il confidential ou public ?

=> Echange éventuel du secret puis la redirection est initiée

Le client est bearer-only

=> Non autorisé (unauthorized)



# Les adaptateurs – cas 2

Cas 2 : l'header authorization est présent

La signature est-elle valide ?

Le token est-il expiré (exp) ?

Vérification de l'issuer (iss)

Vérification de l'audience (aud)

Matching éventuel du not-before





# Liste des adaptateurs

- Adaptateurs Java EE / Servlet containers
- Adaptateur Spring Boot / Spring Security
- Adaptateur NodeJS
- Adaptateur Javascript
- Servlet Filter
- Proxy



# DEMO

- Sécurisation d'un WS REST Java
- Sécurisation d'une application AngularJS



# DEMO - Résumé

- En résumé, nous avons vu
  - Différents types de clients
  - Gestion des rôles



# Sécurité supplémentaire

- Politique de mots de passe
- Support de OTP
- Détection de force brute



# Courtage d'identité / Social Login

- Google, Twitter, Facebook, Github...
- OpenID Connect
- SAML



# Etendre Keycloak

- Personnalisation d'un thème
- Différentes SPI
  - Authentication SPI
  - Action Token SPI
  - Event Listener SPI
  - User Storage SPI

